ISSUE 02 - JUNE 2012

The MagPi™

A Magazine for Raspberry Pi Users

I've got a little something for ya

- *Pi Setup*
- *Interfacing*
- *Pi Dissection*
- *Robotic Arm*
- *Command Line*
- *Music*

Hot off the production line -

The Raspberry Pi at your door

http://www.themagpi.com

ISSN 2051-9990

BY NC SA

The MagPi™

Raspberry Pi is a trademark of The Raspberry Pi Foundation.
This magazine was created using a Raspberry Pi computer.

Welcome to Issue 2 of The MagPi, a community led magazine keeping you up to date with all things Raspberry Pi.

What a month it has been following the release of Issue 1! The interest we received has astounded us, with over 100,000 views by readers as far apart as the US to New Zealand and Russia to South America. We have had positive reviews from the BBC's Rory Cellan-Jones, the Wall Street Journal Tech and many others. Thank you all for your kind words of support and constructive comments.

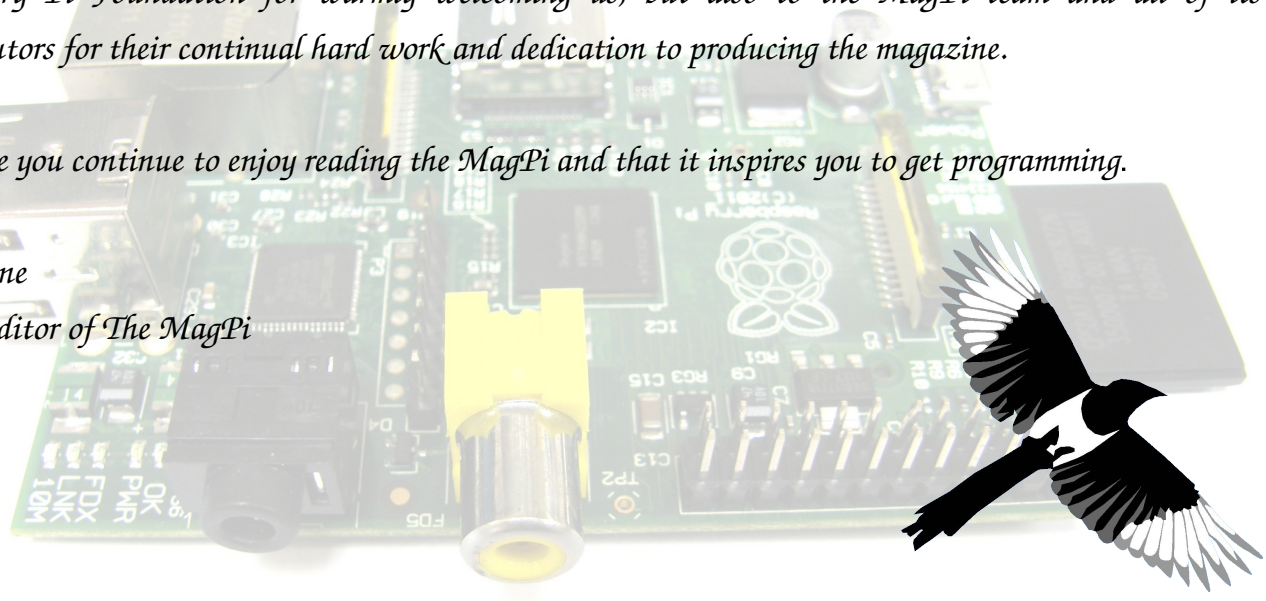
Following the release of Issue 1, some of our readers soon highlighted an error in the logo for JavaScript. We hope that this has not caused too much confusion and wish to clarify that the official logo for JavaScript is as follows: Thank you for drawing this to our attention.

In this issue of the MagPi we return to the Python Pit to give you more tasty tutorials and tips whilst also introducing you to our brand new section, the Command Line Clinic. The Scratch Patch will be returning in Issue 3. We invite you into the world of interfacing, how to install an Operating System on an SD card and setting up the basics on your Raspberry Pi. There is also more on building the skutter robotic arm. If this wasn't enough, we thought we would treat you to a double page spread dissecting the Pi and much, much more.

Before drawing this editorial to an end, I would like to take this opportunity to thank not only the Raspberry Pi Foundation for warmly welcoming us, but also to the MagPi team and all of its contributors for their continual hard work and dedication to producing the magazine.

We hope you continue to enjoy reading the MagPi and that it inspires you to get programming.

Ash Stone
Chief Editor of The MagPi



LITTLE BUNDLES OF JOY

The first orders of Raspberry Pi computers are now reaching the homes of excited owners. Lucky 'Steven from Derbyshire' managed to snare two of them!

Steven a.k.a. 'veryevil' ordered from both Farnell and RS Components, and both orders have been processed. He has loaned one of his Pi's to a friend to work on 'Debian Hard Floating Point'.

We have had many reports of other people starting to get their orders through. Our very own Meltwater now has his - The first of the MagPi Team to get his hands on one. This means we can start to run our own tests and experiments.

With the computers now in the hands of developers and the public, work is rapidly progressing. Add-on boards, ported games, and a phase of software testing has begun.

You can expect there to be a flurry of new discoveries over the next few months. The MagPi will be there to keep you up to speed.

Article by Jaseman

Meltwater's own bundle of joy

Contents

Setting Up The Raspberry Pi P.04
SD Card Setup P.06
Interfacing Projects For Beginners P.10
Eben & Liz Interview Questions P.15
Raspberry Pi Dissection P.16
Skutter P.18
Command Line Clinic P.20
Computer Music P.22
Debian VirtualBox Part Deux P.24
The Python Pit P.26
Feedback P.30
Web Links & Credits P.32

Pi Setup

There are many ways to setup the Raspberry Pi, depending on your needs and the peripherals you wish to use.

Power

The Raspberry Pi uses a micro USB power connector, as used by most modern smart phones.

Micro USB connector

The PSU (power supply unit) must supply 5V and at least 700mA (0.7A). Most plug in PSUs will be marked with their rating. This one will supply up to 2000mA (2.0A).

HP TouchPad PSU

USB

The USB ports on the Raspberry Pi are only suited to low power devices (~100mA). Anything which uses too much current will not function correctly. Therefore, a powered USB hub (a USB hub with it's own PSU) is recommended, so that all devices will have enough power to operate correctly.

Control

Direct control of the Raspberry Pi will be through a mouse and keyboard. Use a standard USB mouse and keyboard or a wireless set using a USB RF dongle. A PS2 keyboard and mouse will need a USB adapter.

Logitech RF Dongle

It should be possible to use a Bluetooth keyboard and mouse using a low cost Bluetooth USB adaptor, but some additional setup will be required.

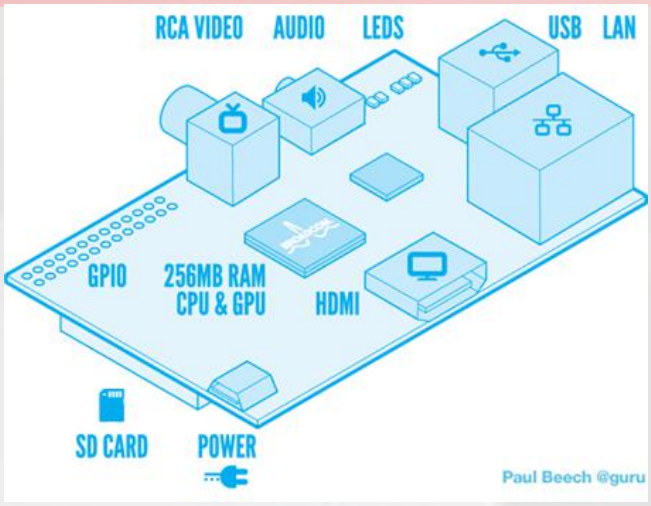
Bluetooth Dongle

Display

TV HDMI Connection

Your display will depend on what you have available to you. Most digital TVs will have an HDMI connection and newer LCD monitors will often have DVI connections (you will need to use an HDMI to DVI adaptor). HDMI to VGA convertors are available but are generally expensive (Note: A passive HDMI/DVI to VGA adapter will not work. The Raspberry Pi does not generate the required analogue signals). The analogue RCA output can connect to older TVs, small screens, or video capture devices (approx. resolution is 640x480, enough for basic use). If both are attached, the Raspberry Pi defaults to HDMI.

HDMI to DVI Adaptor/Cables:
Ensure your DVI connections match as there are several types. Many monitors will only support DVI-D (bottom) with no holes for the four analogue pins DVI-A/DVI-I (top), so an DVI-D adaptor is recommended (this will also fit monitors with DVI-A/DVI-I support).



Tip: Although most connections can be made while the Raspberry Pi is powered on, it is recommended that the display and the SD card are only ever connected or disconnected when you have removed the power first.

Internet Connection (optional)

The easiest option is to use a RJ45 network cable direct to a router or modem, however with some additional setup it is possible to connect to your wifi network with a USB dongle.

Audio Output (optional)

The HDMI connection will transmit digital audio if connected to a TV, however if that is not available you can connect via the analogue audio plug.

Article by Meltwater

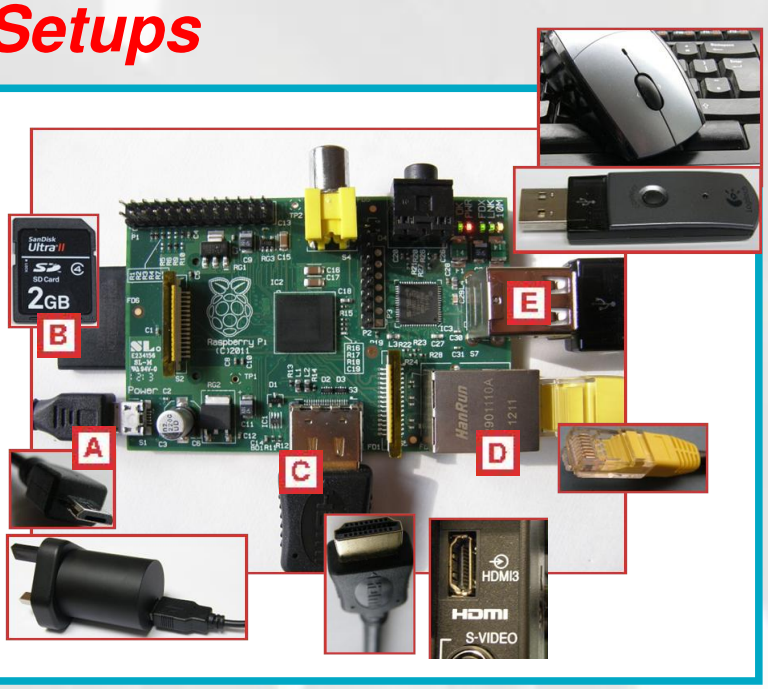
Basic Equipment

1. Micro USB Charger (rated at 5V 700mA minimum), plus micro USB cable if needed.
2. SD Card (2Gb up to SDHC 32Gb), plus suitable card reader.
3. HDMI Cable/Suitable cable for display
4. Compatible Mouse/Keyboard
5. Powered USB Hub (required for high powered USB devices)
6. Network Cable and audio leads if required

Recommended HW see: http://elinux.org/RPi_VerifiedPeripherals

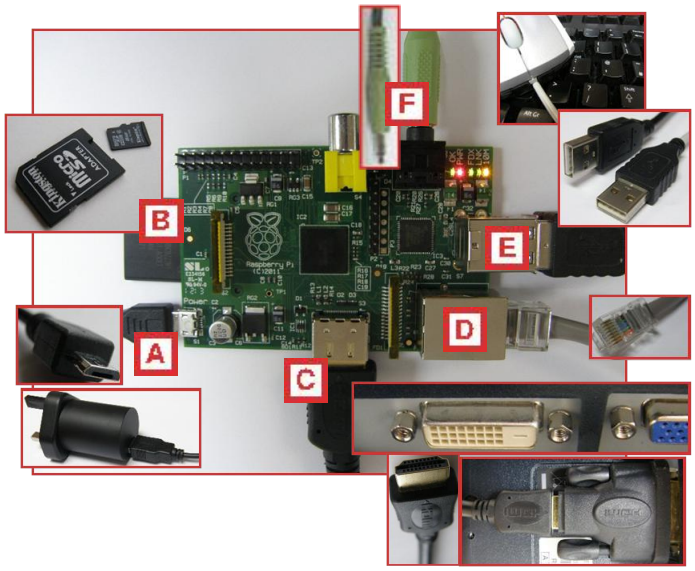
Example Hardware Setups

HDTV Setup



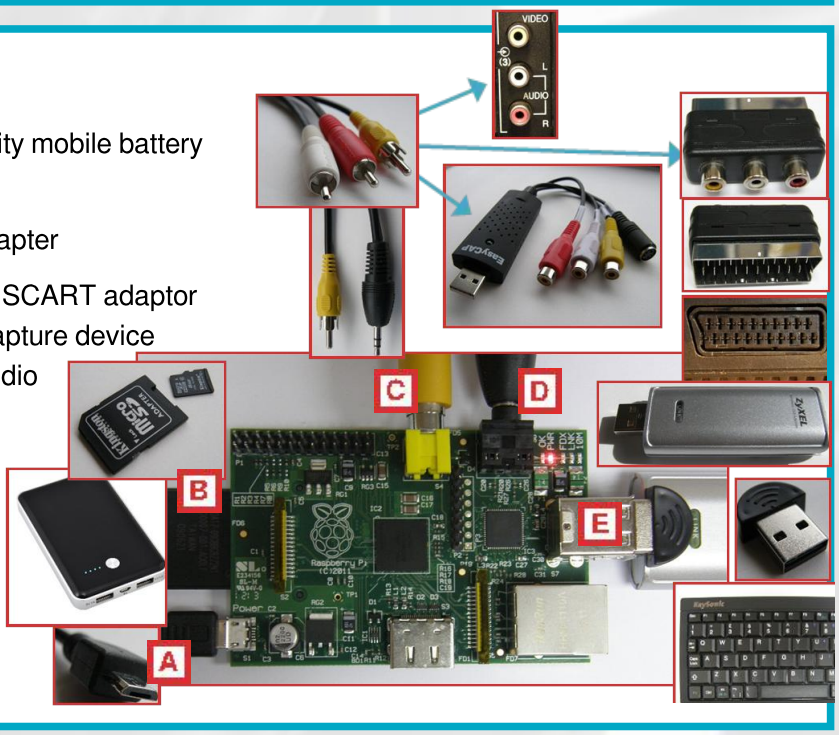
- **A** **Power:** Micro USB charger
- **B** **SD Card:** Standard size SD card
- **C** **HDMI:** Connected to HD TV HDMI input for 1080p & digital audio
- **D** **LAN:** Wire connection to network hub /switch
- **E** **USB:** Wireless RF dongle for mouse and keyboard

LCD Monitor Setup



- **A** **Power :** Micro USB Charger
- **B** **SD Card :** Micro SD Card with Full Size Adapter
- **C** **HDMI :** Connected to HDMI to DVI-D adaptor, then into monitor DVI input for 1080p (no audio)
- **D** **LAN :** Wired connection to network hub/ switch
- **E** **USB :** USB mouse and USB keyboard
- **F** **Audio :** 3.5mm audio out to speakers/headphones

Video Out Setup



- **A** **Power :** Micro USB Charger / high capacity mobile battery (rated output at 5V 700mA or more)
- **B** **SD Card :** Micro SD card with full size adapter
- **C** **RCA Video :** Connected to a TV AV In or SCART adaptor (European-TV connection) / USB video capture device
- **D** **Audio :** 3.5mm audio jack to phono for audio with video
- **E** **USB :** Bluetooth dongle to connect to Bluetooth keyboard / mouse, wifi dongle to connect to wireless network (probably powered by USB hub and additional setup required)

More information see the Wiki Pages: http://elinux.org/RPi_Hardware_basic_Setup

SD Card Setup

Download your preferred Operating System (OS) you wish to run from the Raspberry Pi website:
http://www.raspberrypi.org/downloads

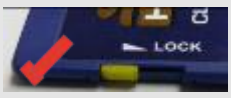
- The download page includes instructions for carrying out file corruption checks (using SHA-1 checksums), recommended to avoid problems.
- Take note of the username and password related to the particular image you have selected. You will need this when you run it.

You will need:
- SD card 2GB or larger (depending on OS and other requirements)
- Suitable SD card reader (some built in card readers have issues so a USB card reader is recommended)
- PC running Linux/Windows or even a Raspberry Pi! *(If using a Raspberry Pi use a powered USB hub with a SD card reader to write the image to a new card. Warning, it will be very slow this way!)*

The following instructions apply to the `2013-02-09-wheezy-raspbian.zip` file, but the steps should be similar whichever package is selected.
Once you have written to your card, you are ready to boot your Raspberry Pi (see guide later on).

To reinstall the Operating System or to select a different one, simply repeat the process.

NOTE:
Ensure the SD card write "Lock" is disabled!



Writable

Read Only

SD CARD SETUP WITH WINDOWS

Extract the image file to your hard drive
For ZIP Files: Explore using Windows and copy and paste contents out to your hard-drive.
For tar/bz2 file, use PeaZip www.peazip.org.

You'll need at least 2GB of space to extract the file. You should end up with a .img file (this contains all the parts of the Operating System which needs to be written to the card).

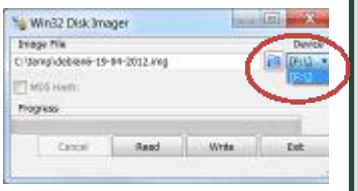
Write the image to the SD card
You will need to use a program such as **Win32DiskImager** (available from http://www.raspberrypi.org/downloads). This program will write the image structure required for the Raspberry Pi to boot from the SD card.

Insert your SD card into the card reader. Remove any other drives/devices you may currently have attached to avoid wiping the wrong device by mistake.

Open up the Win32DiskImager program

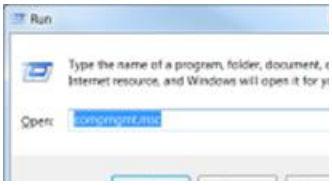
a) Select the .img file you extracted previously by clicking on the small folder icon and browsing to where you extracted it.

b) Select the device letter which relates to your SD card. If in doubt double check this letter in Windows Explorer as the process will remove all data on whatever drive you select.

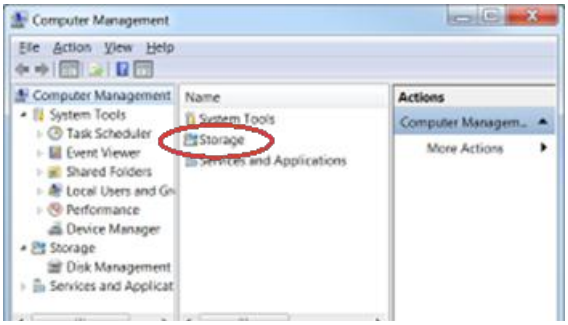


c) Click the `Write` button and you will see the program start to write to the SD card. Note, you can also use `Read` to backup your SD card!

To see the result you can open up the Disk Management program built into Windows.

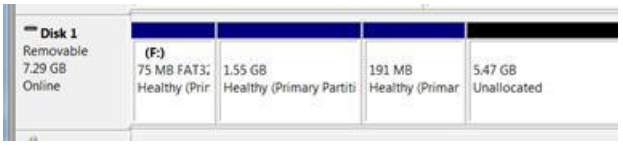


a) Press `Windows key + R` or select `Run...` from the Start Menu.



b) Type `compmgmt.msc` and press OK

c) You will see the above window appear. Select the Disk Management component, then Storage and you should see your drives, including the SD card with several partitions (most likely 3):
- boot: viewable in Windows
- OS System: not viewable in Windows
- OS Swap drive: also not viewable in Windows
- ...plus any unallocated space at the end (see guide on resizing this).



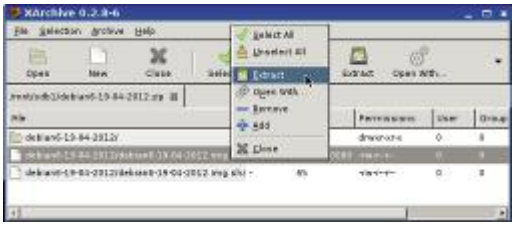
SD CARD SETUP WITH LINUX

Note: These steps should apply to most common versions of Linux. Screenshots are from RacyPy (Puppy Linux, MagPi - issue 1)

Extract the image file to your hard drive

Locate your downloaded OS file in your file manager. Remember you'll need at least 2GB of space to extract the .img file.

GUI programs such as XArchive (pre-installed on RacyPy) will allow you to locate and extract the .img file directly.



Alternatively use this command in a terminal window:

```
unzip filename.zip
```

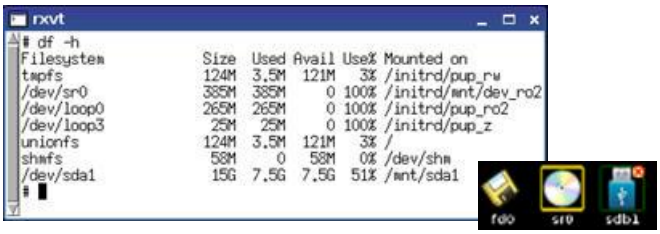
The "filename.zip" should match the source file. Note, it will take a while to extract the files.

Open a terminal window:
You can normally open a terminal window directly from the file manager i.e. RacyPy/Puppy ROX-Filer **right-click**, select **Window -> Terminal Here**
In Raspbian open the File Manager and press **F4**

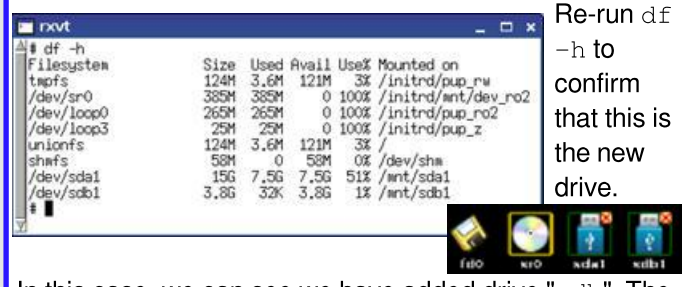
Identify your SD card device name

Before you plug in your card reader and SD card take note of any devices you may have already plugged in or installed. You can usually see icons on the desktop, or run this command in a terminal window:

```
df -h
```


rxvt
df -h
Filesystem Size Used Avail Use% Mounted on
tmpfs 124M 3.5M 121M 3% /initrd/pup_rw
/dev/sr0 385M 385M 0 100% /initrd/mnt/dev_ro2
/dev/loop0 265M 265M 0 100% /initrd/pup_ro2
/dev/loop3 25M 25M 0 100% /initrd/pup_z
unionfs 124M 3.5M 121M 3% /
shmfs 58M 0 58M 0% /dev/shm
/dev/sda1 15G 7.5G 7.5G 51% /mnt/sda1
#
fd0
sr0
sdb1

Attach your SD card reader with the target SD card inserted. On RacyPy/Puppy you will see an extra USB icon has appeared (or browse through the file manager). Open it (this will "mount" it).


rxvt
df -h
Filesystem Size Used Avail Use% Mounted on
tmpfs 124M 3.6M 121M 3% /initrd/pup_rw
/dev/sr0 385M 385M 0 100% /initrd/mnt/dev_ro2
/dev/loop0 265M 265M 0 100% /initrd/pup_ro2
/dev/loop3 25M 25M 0 100% /initrd/pup_z
unionfs 124M 3.6M 121M 3% /
shmfs 58M 0 58M 0% /dev/shm
/dev/sda1 15G 7.5G 7.5G 51% /mnt/sda1
/dev/sdb1 3.8G 32K 3.8G 1% /mnt/sdb1
#
fd0
sr0
sda1
sdb1

Re-run `df -h` to confirm that this is the new drive.

In this case, we can see we have added drive "`sdb`". The partition "`sdb1`" is 3.8Gb and 32K is used. This matches what we expect to see for this SD card.

"Unmount" the partitions

You now need to "unmount" the device (this closes the drive so that the program is able to write to it).
In RacyPy/Puppy you can right-click on the drive icon and select "**Unmount sdXY (if currently mounted)**".
If you've used the card before, you may have multiple partitions, so ensure you "unmount" them all.

Alternatively use this command in a terminal window:

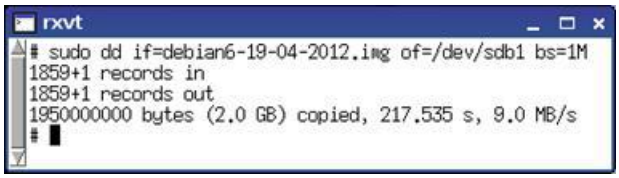
```
umount /dev/sdb1
```

where `sdb1` is the name of the device partition above.

Write the image to the SD card

Locate the directory which contains the extracted .img file and open a new terminal window.

Write the image with the following command (where `sdX` should match your device name - without the number at the end).
Take care on this step since this command will overwrite all the data on the drive - so ensure you get the right drive.




```
sudo dd if=filename.img of=/dev/sdX bs=1M
```

`sudo` – allow permissions to write to the SD card device
`dd` – disk-to-disk program
`if` – input file or source drive
`of` – output file or target drive
`bs` – size of blocks to use during copy (1M is fine)

Advanced Tip: You can also use `dd` to backup your SD card:

```
sudo dd if=/dev/sdX | \
split -d -b 2000m - image.split
```

To restore the split image (splitting avoids oversized files):

```
sudo cat image.split* | dd of=/dev/sdX bs=1M
```

Writing the image to the SD card will take some time (lookout for the card reader activity light flashing). Linux will report when the process has completed.
Remove and re-insert the SD card to confirm there are now two new partitions visible on the card (**sda1 Boot** and **sda2 Operating System**). There is also a third hidden swap partition also.

If you run into problems creating your SD card additional help and instructions are available on the Raspberry Pi Wiki (Mac instructions are also available).
http://elinux.org/RPi_Easy_SD_Card_Setup
If you have general problems, take a look at the trouble-shooting section.
http://elinux.org/R-Pi_Troubleshooting

Article by Meltwater

Resize Partitions

In order to maximise the use of an SD card, it is recommended the main partition of the card is enlarged to fill up the remaining unallocated space on the card.

To do this, you will need a partition management program. The Linux program GParted is recommended as most software available for Windows cannot manipulate the Linux formatted partitions you now have on the SD card. If you don't have Linux installed, you can use a LiveCD or VM (Virtual Machine), such as Puppy/RacyPy or Debian which were covered in The MagPi issue 1.

There is even a specific GParted LiveCD, available with instructions on how to use it:

`http://gparted.sourceforge.net/livecd.php`

If GParted is not already installed on your Linux system, it can be installed with the following command in a terminal window:

```
$sudo apt-get install gparted
```

Running GParted

Start GParted from the application menu, (for Puppy/RacyPy it is located in the System section).

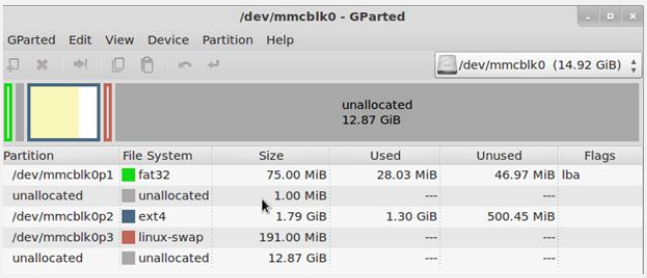
Alternatively, you can start the program using the terminal prompt by typing:

```
$sudo gparted
```

If prompted, enter your user password to give the program admin rights.

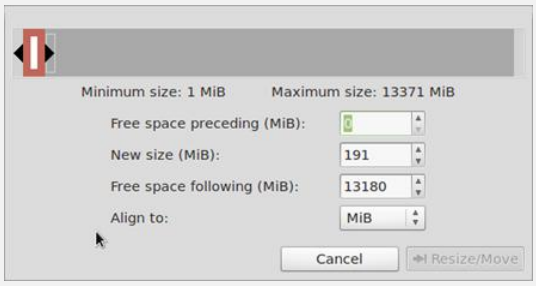
When the GParted utility scans the drives and loads, click the dropdown box in the upper right hand corner and you should see an entry which matches your SD card. If you are unsure, add/remove the card and rescan by pressing CTRL+R, or via the "GParted" menu/ "Refresh Devices" until you determine which one it is.

Disclaimer: We are going to be altering disk partitions, so you MUST make sure you've selected the right disk (card in our case). If your screenshot differs vastly from that below, stop and ask yourself some stern questions before continuing. We are not responsible if you end up deleting your installed system. Remember always, you are running this as a super user. Please be careful.

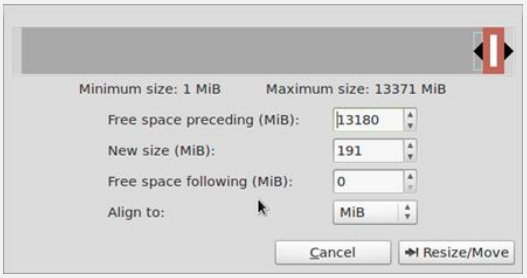

/dev/mmcblk0 - GParted
GParted Edit View Device Partition Help
/dev/mmcblk0 (14.92 GiB)
unallocated
12.87 GiB
Partition File System Size Used Unused Flags
/dev/mmcblk0p1 fat32 75.00 MiB 28.03 MiB 46.97 MiB lba
unallocated unallocated 1.00 MiB --- ---
/dev/mmcblk0p2 ext4 1.79 GiB 1.30 GiB 500.45 MiB
/dev/mmcblk0p3 linux-swap 191.00 MiB --- ---
unallocated unallocated 12.87 GiB --- ---

You can now see the layout of your SD card. You should see 3 partitions in the top part of the window (Green, orange and blue in the above screenshot). We have a small boot partition, your 1.8GB home partition and a swap partition. What we need to do is to move that swap partition all the way to the right, and then stretch out the home partition to make it bigger.

Move the swap partition. Right click on the third partition and choose Unmount/SwapOff. Then right click it again and choose Resize/Move. You'll see this screen:



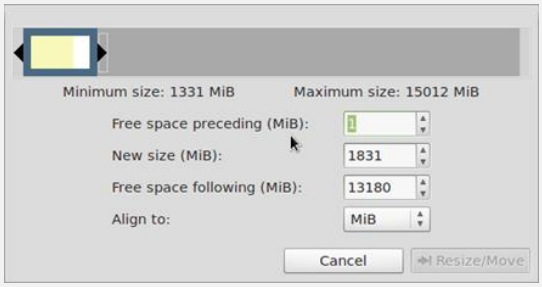
In the Free space following (MiB) type 0 and press enter, or grab hold of the partition (at the top) with your left mouse button, and drag it all the way to the right:



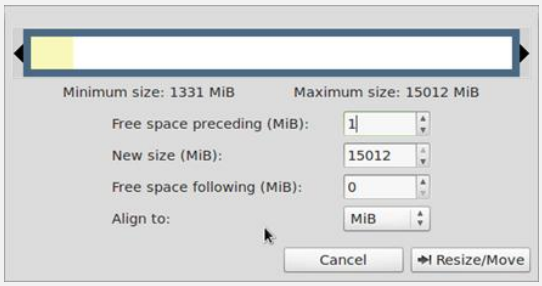
Click Resize/Move. A warning will pop up

stating that this may cause your operating system to fail to boot. Ignore this and click Ok.

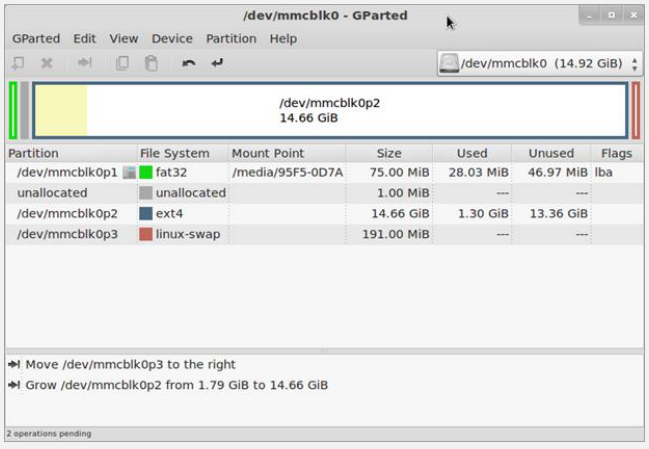
Stretch out the home partition. Right click on the second partition in the top of the window (the biggest partition), and again choose Unmount. Right click it again and choose Resize/Move. You'll see this screen:



Grab the little arrow to the right of the partition with your left mouse button, and stretch it all the way to the right:



Choose Resize/Move.
Your screen should now be like this (please note that I have a 16GB SD card here, you may have one of a different size):


/dev/mmcblk0 - GParted
GParted Edit View Device Partition Help
/dev/mmcblk0 (14.92 GiB)
/dev/mmcblk0p2
14.66 GiB
Partition File System Mount Point Size Used Unused Flags
/dev/mmcblk0p1 fat32 /media/95F5-0D7A 75.00 MiB 28.03 MiB 46.97 MiB lba
unallocated unallocated 1.00 MiB --- ---
/dev/mmcblk0p2 ext4 14.66 GiB 1.30 GiB 13.36 GiB
/dev/mmcblk0p3 linux-swap 191.00 MiB --- ---
Move /dev/mmcblk0p3 to the right
Grow /dev/mmcblk0p2 from 1.79 GiB to 14.66 GiB
2 operations pending

Now we need to write the changes to the card.
Open the Edit menu and choose Apply All Operations. You will now see a confirmation box. Click Apply.

Wait for GParted to complete the operation (this may take a minute or two, depending on the speed of your SD card). When it's finished, click OK and the close GParted.

You may now safely remove the SD card from your doner computer and put it back in your Raspberry Pi. Boot your Raspberry Pi and enjoy your new larger home partition!

Article by Pete / Meltwater

First Boot

You have all your hardware, you've connected those wires (all except the power), you've prepared your SD card and you are ready to go. Remember, keep it simple and only connect the minimum hardware at first until you are happy everything is running smoothly (ideally the keyboard/mouse should be connected through a powered USB hub if available).

Double check everything, switch on your display and now plug in the power! You should see the red PWR LED light up and within the count of 10 the OK light should start to flash activity. On your display you will see lots of text scrolling down the screen. This is the Raspberry Pi initialising. The first time it will take longer as it configures itself and resets. If you are using a TV and the text doesn't fit on the display, try changing the aspect ratio settings (i.e. widescreen, 16:9, 4:3, etc) or if you still have issues check the wiki (http://elinux.org/R-Pi_Troubleshooting).

You should end up with a prompt similar to:

```
Debian GNU/Linux
raspberrypi login:
```

You'll need to enter the username and password for the particular distribution you downloaded (for Raspbian, login with the user `pi` and password `raspberry`).

Finally, to start a new graphical session type `startx`. To power down, select shutdown from the bottom right corner in the graphical session or type `sudo shutdown -h now` from the command terminal.

If you have made it this far congratulations. The proof of the Pi really is in the eating.

IN CONTROL

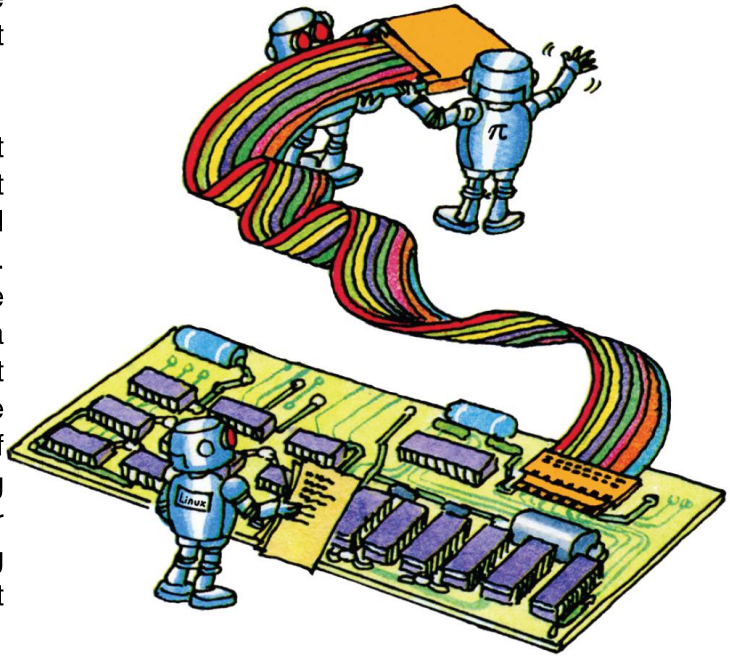
INTERFACING PROJECTS FOR BEGINNERS

BY DARREN GRANT

If you want a change from playing games or browsing the internet, what else is there that you can do with a computer? What I would like to share with you is the tremendous fascination of using a computer for control. You can get the computer to switch things on and off, respond to events and take a variety of measurements. This is known as 'interfacing'.

If you feel your knowledge of electronics is rather limited, don't despair. In this series we are not assuming that you have more than a basic understanding of electronics. The aim is to explain the basics and provide details of relatively simple hardware and basic software to get started.

The Raspberry Pi provides an ideal way to begin experimenting. Not only does the low cost of the Pi mean that there is no need to worry about damage to an expensive computer, it also comes ready equipped with the connections to get started right away.



As you will have no doubt probably figured out already computers are all about In's and Out's. Input something on the keyboard or click with a mouse and get something out on the screen or sound out of the speakers. Interfacing is simply finding other ways of getting information in to and out of the computer.

The first thing to understand is that the basic building blocks of a computer are made up of millions of switches called transistors. Computers see everything as either on or off so when you click the mouse button the switch is on and when you let go it is off. The computer has no idea how hard the button has been pressed. This is known as a digital binary system: when a switch is on it is represented by a '1' (one) and when it is off a '0' (zero). Signals that are variable such as measuring temperature are called analogue. As they are more complicated than just a simple on or off state, extra equipment is needed so the computer can read them.

For now we will stick with a simple digital interface. The Raspberry Pi has a set of connections called the GPIO (General Purpose Input Output) ports. GENERAL PURPOSE means that they have not been assigned a specific purpose, unlike for example the speaker connection that has a well defined purpose. The connections on the GPIO port can be used for whatever purpose you like. Each connection can be used as an INPUT where the computer is waiting for an event to happen such as a switch being pressed or an OUTPUT where the computer is sending a signal out for example to switch a light or buzzer on and off.

The GPIO ports on the Raspberry Pi are low powered so an expansion board is recommended to provide higher powered signals that can be used for controlling things like lamps, buzzers and motors. A GPIO expansion board also provides some protection to the Raspberry Pi just in case something is connected incorrectly.

Don't worry if you don't have a GPIO expansion board at this stage and you are raring to get going with your first experiment. Providing you take care we can start with a simple circuit using a switch and a LED (Light Emitting Diode). ►

IMPORTANT
Before connecting anything to the Raspberry Pi please be aware that incorrect connections could cause damage. Please take care.

Switch Types

An electrical switch is used to make or break an electrical circuit. Switches are binary devices: they are either completely on (closed) or completely off (open) making an ideal starting point for our interfacing experiments. There are many different types of switches, the simplest type being one where two electrical conductors are brought in contact with each other by manually moving a mechanism.

There are essentially two common types of switch, latching switches and momentary types. A latching switch stays open or closed when activated, for example a light switch; when you switch the light on or off the switch stays in the same position until you switch it again. A momentary switch on the other hand will only operate while pressure is being applied and will return to its previous state when released. Momentary switches can be found on a game controller where the control only responds while the button is being pressed. A momentary switch wouldn't make a very good light switch as you wouldn't want to have to keep the button pressed to keep the light on. The electrical symbol for a simple momentary push switch is this.

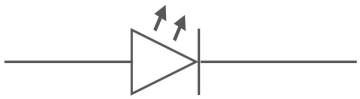
We will use a small momentary push button switch for this experiment, known as a normally open tactile switch.

LEDs

LEDs (Light Emitting Diodes) can be found everywhere, commonly used as indicator lights on most electronic equipment such as the standby light on your TV.

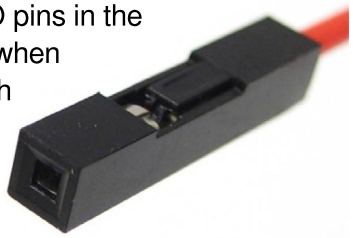
Due to the small amount of power needed to make a LED light up, we can safely use them in our experiment without damaging the Raspberry Pi.

The symbol for a LED looks like this.



To save soldering wires together, a breadboard and a selection of jumper wires is highly recommended. It makes connecting things together very easy and flexible. The grid reference numbers are laid out on a small 270 point Tandy breadboard. If you have a bigger board that will work just as well but the reference numbers may not match so will need to be modified if your board has a slightly different grid layout.

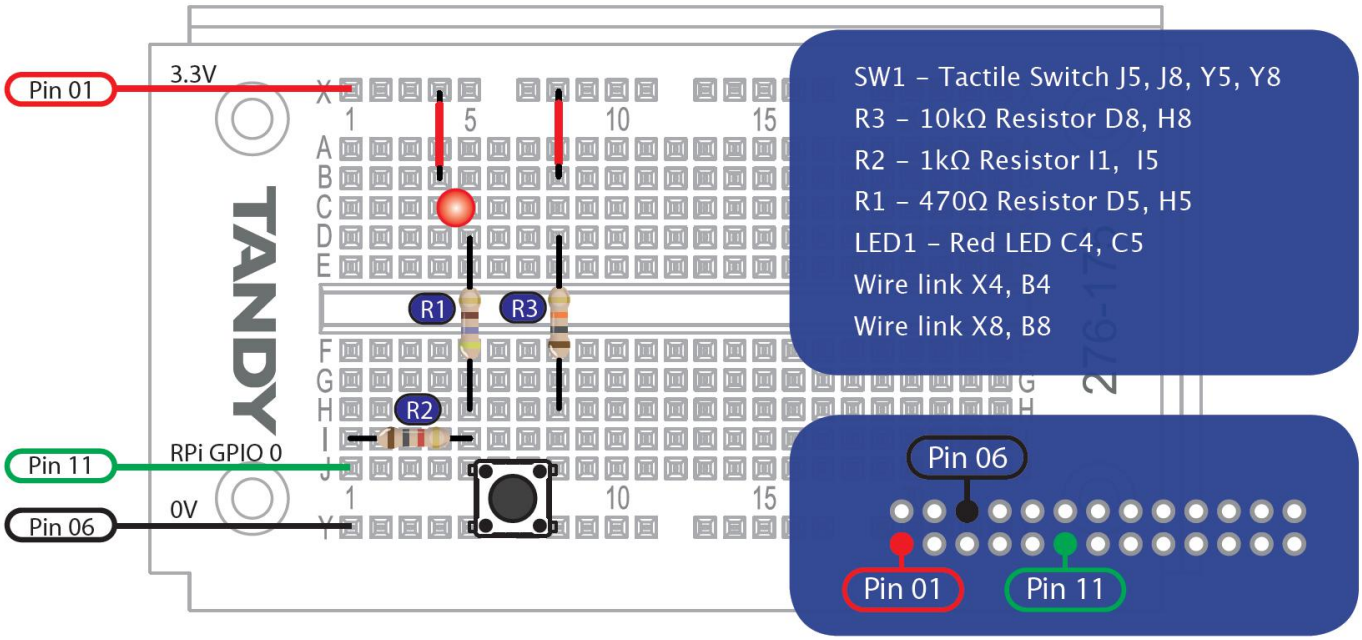
Place the components on the breadboard using the illustration as a guide. If your tactile switch has inward pointing legs like the one shown in the photo on the top left, use a small pair of pliers to straighten them. The switch needs to be inserted into the breadboard so that the legs are on the left and right edges. It is also important that the LED is connected the correct way round. The positive terminal called the anode is the longer leg and should be connected to point C4. Once all the components are in place connect the circuit to the Raspberry Pi using 3 female to male jumper wires, taking extra care to connect the correct terminals.



The Raspberry Pi GPIO pins in the illustration are shown when looking at the board with the GPIO header in the top left corner, as shown in the photo in the Pi Dissection article on page 16.

Having connected everything together, power up the Raspberry Pi and try pressing the button. The LED should light up. If it doesn't work check that all the components and wires are connected correctly, paying particular attention to the LED. If the LED stays on without pressing the button then the switch is probably connected incorrectly and will need to be turned 90°.

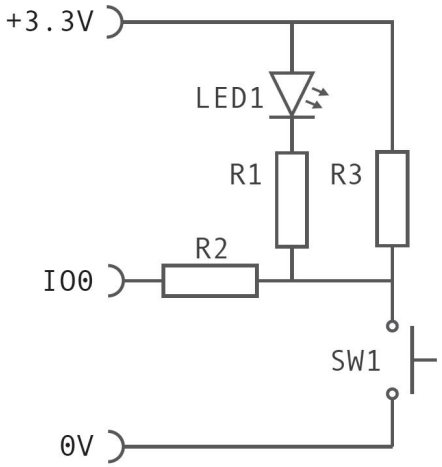
Continued over page...



Circuit Description

Because the Raspberry Pi GPIO port is not protected with a buffer, we have included a 1KΩ (Ohm) resistor (R2) between the switch and the GPIO pin as a precaution to prevent damage if the GPIO port is accidentally set as an output. Our tactile switch is connected to the 0V or ground line so that when it is pressed the circuit is completed between the GPIO pin and zero Volts.

The LED along with a 470Ω current limiting resistor (R1) is connected to the 3.3V positive rail. Pressing the switch completes the circuit between the LED and 0V causing it to light up, this provides a visual indication when the switch is pressed. Finally we have a 10KΩ pull-up resistor (R3).



Digital Logic States

As mentioned earlier, computers see the world as a series of zeros and ones. The Raspberry Pi uses 3.3V CMOS logic meaning that a binary 1 is created by applying a voltage of 3.3V to an input.

When the voltage is present it is referred to as a logic HIGH state. Alternatively when the GPIO pin is connected to 0V a binary 0 or logic LOW state is created. In our switch example pressing the switch creates a LOW state by connecting the IO port to 0V. ►

Shopping List

COMPONENTS
1x 3mm Red LED (standard brightness)
1x 470Ω Resistor
1x 1kΩ Resistor
1x 10kΩ Resistor
1x Miniature PCB Tactile Switch

ACCESSORIES
1x Breadboard
3x Male-female jumper wires
1x Selection of short jumper wires

TOOLS
If you don't have them already then a set of small long nosed pliers for bending component leads and wire cutters will make construction easier.

Digital logic in fact has three possible states known as Tri-State logic. We have already discussed HIGH (+3.3V) and LOW (0V) states but we also have a third state that is called FLOATING. Basically floating means the state is not clearly defined. Attempting to determine the status of a connection that is floating may result in unpredictable results. You might be thinking that if the pin is not connected to a voltage it must be low: the problem is we can not guarantee that we will get a reliable LOW signal by simply relying on the absence of a voltage on the pin.

To illustrate the problem, imagine that we hold a piece of ribbon between two gate posts, the two gate posts would be connected by the ribbon. Now if we let one end of the ribbon go and allow it to fall to the ground the two posts are no longer connected. But what happens when the wind blows? The ribbon will flap around where sometimes it will be in contact with the ground and other times it will touch or even become entangled with the other post. To prevent the ribbon moving in the wind we would tie it either to the ground or the post so it wouldn't move unless we wanted it to. We need to do the same thing with a logic circuit, to avoid unpredictable behaviour. We tie the connection to either 3.3V or 0V using what is known as a pull-up or pull-down resistor that effectively creates a default state. In this experiment we have used a pull-up resistor so the IO port will always be connected to +3.3V making it the default state. In our example when reading the state of the IO pin it will always be high until we press the switch that will change it to low.

Software

Having completed the circuit construction we now have to get the Raspberry Pi to recognise the switch. To do this we need to create a software program that will instruct the Pi to monitor the switch status.

We will be using the Python programming language to write our program. It is assumed that you are using the standard Raspbian operating system from the Raspberry Pi website. If you have not used Python before take a look at the guide to editing and running Python programs on page 24.

The first thing we need to do is install a Python package that adds the capability to Python to control the GPIO pins on the Raspberry Pi. Download the Python package from http://pypi.python.org/pypi/RPi.GPIO the latest version at the time of writing this article is RPi.GPIO-0.5.0a. If the file version downloaded is different then the instructions below will need to be modified to use the version number you have downloaded.

We will use the terminal to install the package, to open the terminal go to the main menu, select accessories and click on LXTerminal which will open a new window with a command prompt. The file is compressed and archived. To install it we will need to decompress and unarchive it by typing the following commands.

```
gunzip RPi.GPIO-0.5.0a.tar.gz
tar -xvf RPi.GPIO-0.5.0a.tar
```

We now have a new directory called RPi.GPIO-0.5.0a that we need to go into and run the setup script. Type the following commands to complete the installation.

```
cd RPi.GPIO-0.5.0a
sudo python setup.py install
```

You may be asked for your password when running the install script. If you have not changed the password the default is 'raspberry'.

Now that the package installation is complete we can write a program to monitor the switch and write something to the screen each time it is pressed. Before we start programming, type the following command at the command prompt to return to the home directory.

```
cd ..
```

Continued over page...

Python v2.7 Program

Create a new text file named mybutton.py and enter the following program.

```
#!/usr/bin/python
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.IN)

while True:
    mybutton = GPIO.input(11)
    if mybutton == False:
        print "giggle"
        time.sleep(.2)
```

Program Description

We start by including the time and RPi.GPIO packages in our program so we can make use of their functions by using import.

Next we configure GPIO pin 11 on the Raspberry Pi as an input so it can detect the switch.

By using 'while True' we create a never ending loop so that everything below this will be repeated until we choose to stop it.

Our program keeps checking the status of pin 11 that will always be True (High) while the button is not being pressed. As soon as the button is pressed the GPIO pin goes low and our result will be False, at which point we print out the word "giggle" on the screen. Because the program runs much quicker than we can press and release the button we add a small 0.2 second delay before checking the button status again so that the screen does not fill up with giggles.

To start the program type the following command into the terminal window.

```
sudo python mybutton.py
```

When you have had enough giggles press the Control+C keys to stop the program.

Counting

Try the following program to count the number of times the button is pressed.

```
#!/usr/bin/python
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
GPIO.setup(11, GPIO.IN)
count = 0

while True:
    mybutton = GPIO.input(11)
    if mybutton == False:
        count = count+1
        print "count", count
        time.sleep(.2)
```

Conclusion

We have covered how to make the computer respond to a switch circuit and written a simple program. Now that you have mastered the basics how about you try and improve the program. For example make a timer program that will start and stop counting when pressing the button. Why not send in your programs and questions to in.control@themagpi.com or get involved on the official Raspberry Pi forum. Maybe we can include some of them in the next instalment. ●

Tandy®
www.tandyonline.co.uk

This article is sponsored by Tandy.
Look out for our new web site coming soon: where you will find all the components you need for your Raspberry Pi interfacing projects and much more.

STOP THE PRESS!

EXCLUSIVE INTERVIEWS

Do you have Raspberry Pi questions that you just can't find the answer to?

Have you searched and searched the forum to no avail?

Don't fret, the MagPi is here to help!

We are pleased to announce that we are lucky enough to have an exclusive opportunity to interview both Eben and Liz, the King and Queen of the Raspberry Pi.

If you have a question that you want us to put to those at the top of the Raspberry Pi Foundation, please contact us at editor@themagpi.com and look out for this exciting article in the near future in The MagPi.

Pi Dissection

2V5 Pin 5 of RG3 (LP2980 Voltage Regulator) used for Composite Video Out

USB Gnd (TP2)
Ground pin of Micro USB Power Supply

1V8 Pin 2 of RG1 (NCP117-1V8 Voltage Regulator) used for SDRAM & CPU Core.

DSI (Display Serial Interface) (S2)
This will support direct connection to add-on screens (when available)

SC Card Socket (S9)
On power up, the Raspberry Pi will load the operating system directly from a suitably prepared SD card.

Micro USB Power (S1)
Only Pin 5 Ground and Pin 1 5V are connected (no data or ID)

PCB — Printed Circuit Board
The components are connected through 6-layers of tracks sandwiched together and connected through "vias" (holes through the board). One layer, called the "Ground Plane" (0V) is used to shield the densely packed signals, and separate high-frequency data from sources of interference.

5V In (TP1)
5V supply after the 1A1 Polyfuse (F3 fitted on rear)

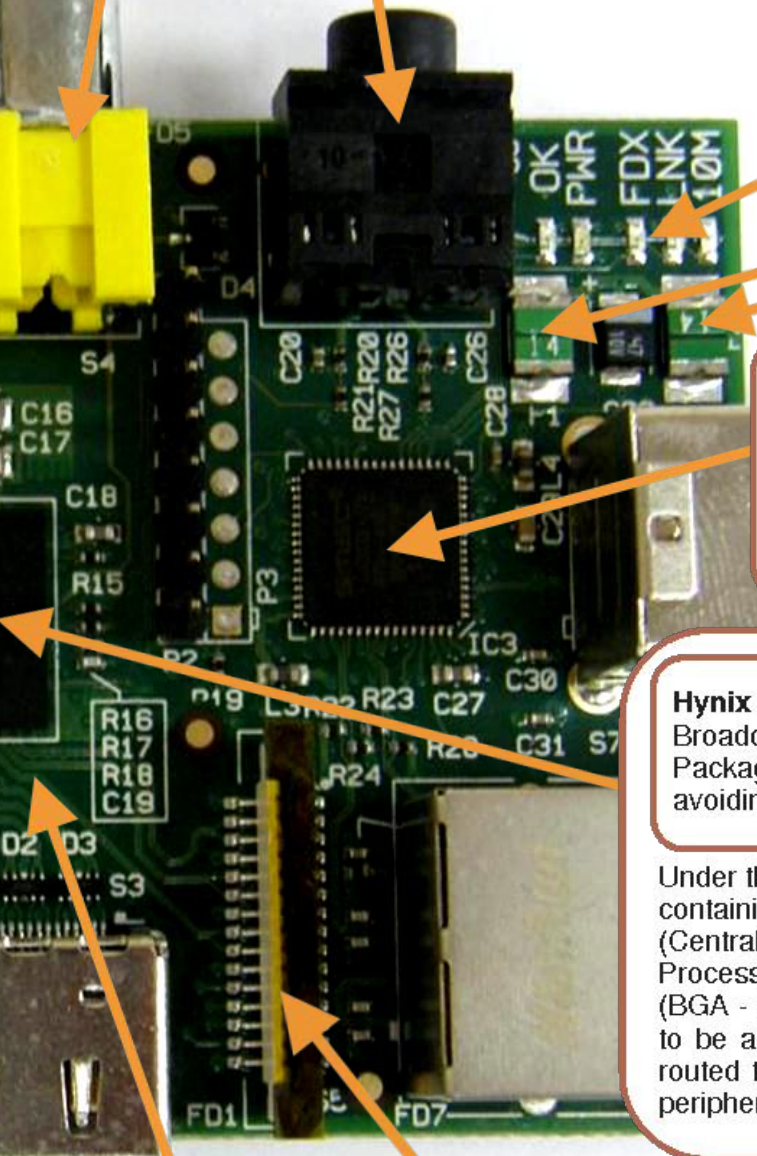
3V3 Pin 3 of RG2 (NCP1117 Voltage Regulator)

Tip: Engineers often use 3"V"3 notation in electrical drawings (schematics) to indicate 3.3V or 1K2 (=1200) to ensure the decimal point is clearly marked (mistaking 33V instead of 3.3V would be a big problem).

Composite Video Out & Analogue Audio
Signals are generated internally by the SoC using on-board DAC (digital to analogue converter) for video and a smoothed PWM (pulse width modulation) signal for each right and left audio channels.

LED Lights (D5-D9)
Status OK (D5 Green) [BCM GPIO16]
Power On (D6 Red) [3.3V Supply]

Full Duplex (D7 Green)
Link (D8 Green)
100M (D9 Yellow) -
100Mbps when lit!



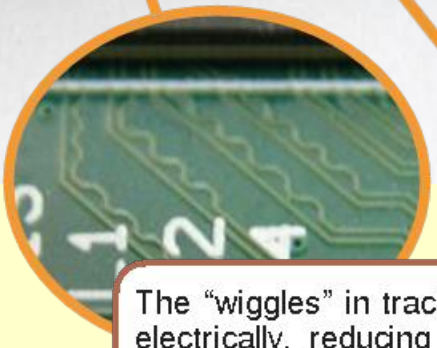
140mA Polyfuses (F1 & F2)
Limit the current drawn by USB devices.

LAN (Local Area Network) Chip (IC3)
The LAN9512 is connected via the SoC's USB interface and provides 10/100Mbps wired LAN and dual USB connections. Model A of the Raspberry Pi, will have the SoC's USB connection connected directly to a single USB socket.

Hynix 256Mb SD-RAM - Wired directly on top of the Broadcom BCM2835 chip using PoP (Package on Package) method, this reduces the required footprint by avoiding 68 pin connections related PCB tracks.

Under the RAM, is the **Broadcom SoC (System on Chip)**, containing 700MHz ARM11 (ARM1176JZF-S) CPU (Central Processing Unit) and VideoCore IV GPU (Graphical Processing Unit). The SoC's 300 pins are arranged in a grid (BGA - Ball Grid Array) allowing the physical size of the chip to be as small as possible. Each of the required pins are routed through one of the 6 PCB layers to the surrounding peripherals.

CSI-2 (Camera Serial Interface) (S5)
This will allow connection to add-on camera modules (when available). The GPU is potentially able to handle up to 40MPixels stills and 1080p 30 fps (frames per second) video capture.



The "wiggles" in tracks, ensure signals are matched electrically, reducing interference and signal delay. This is particularly important for high speed video data and HDMI signals.

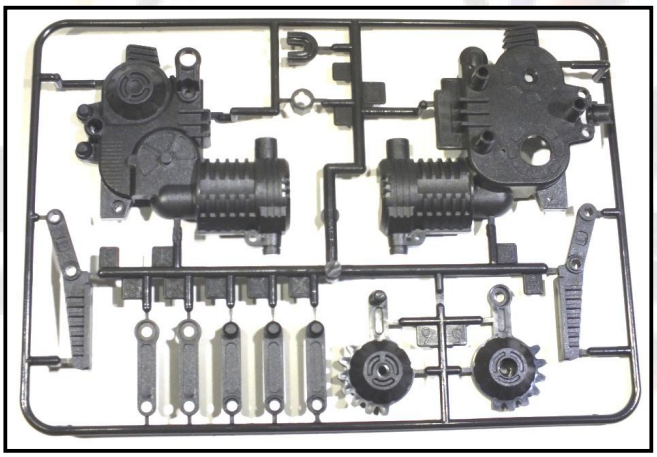
Information based on Raspberry Pi Rev1 board and given as a guide.
Compiled by Meltwater

SKUTTER

By tzj

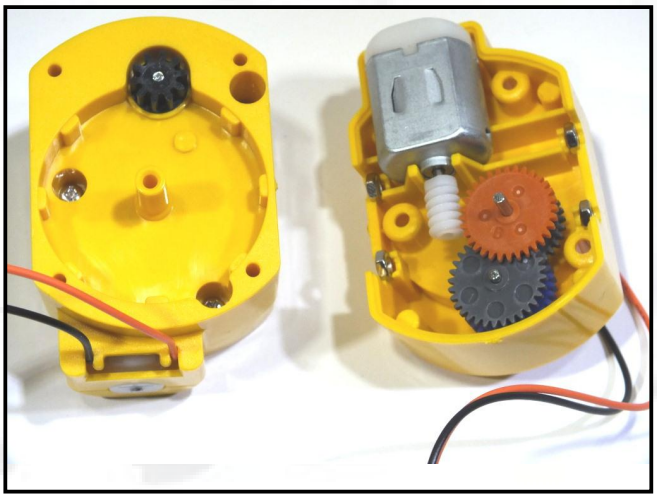
Bodge N Hackitt has been unable to provide us with part 2 of his Skutter project this month, due to exams and family commitments. But he will be back!

In the meantime, tzj has bravely taken up the challenge of making his own Skutter. He purchased the OWI robotic arm kit, and is going to share with us his experience of putting it together. So you can consider this article as 'Skutter Part 1.5'



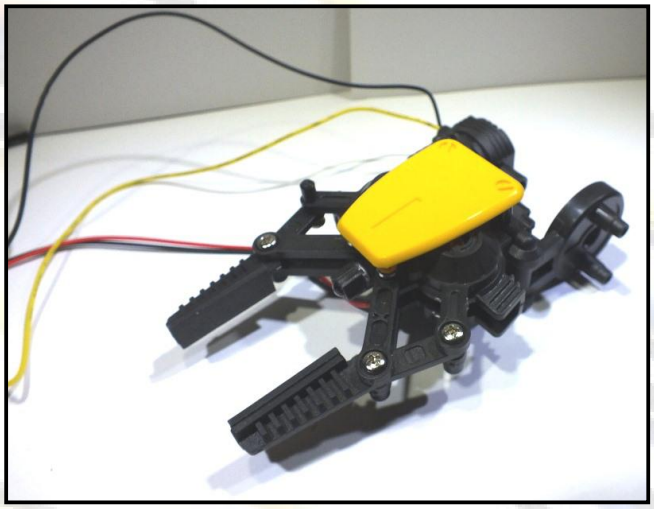
The plastic airfix-like parts are a little tough to get through when clipping parts out, so adult supervision is advised.

The nuts for the gear boxes are a bit of a fiddle to put in but so long as you have long nosed pliers, you should be ok.

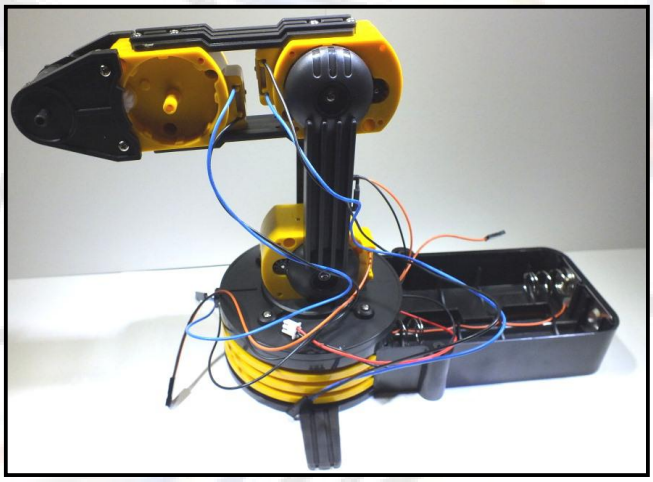


Wires at the motors are really hard to push

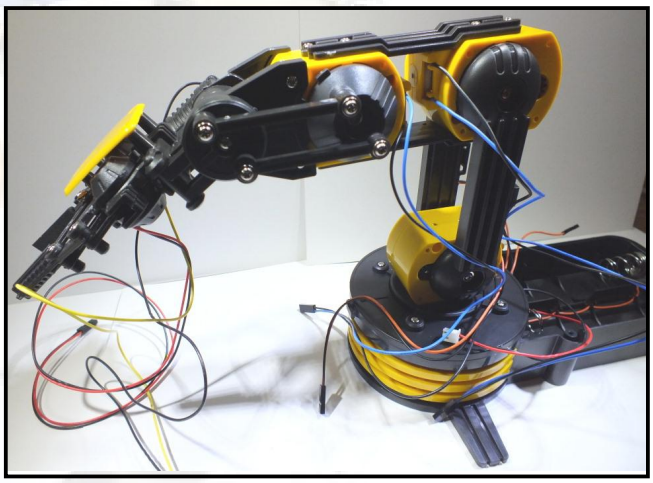
into the clips. A thin flathead screwdriver does the trick.



To make my assembly easier I also built the gripper gear assembly at the same time as the other gear assemblies.



The base was fairly easy to assemble, though wire routing is a challenge in places.



Note - 3V is taken from the middle (large) battery terminal. So I suggest you don't go out and get a 6 volt lantern battery to save money just yet, like I did.

The software works but isn't optimised for larger screens. I had to download the 64-bit software and driver for it to work.

I had my next door neighbour's 4 year old son playing with the arm; once he got used to using it properly (rather than attempting to manually move it), he had a lot of fun. However, he did have better control over it using the mouse rather than the keyboard.

After getting out my scales, the arm's weight with the batteries in is 1.155Kg, and the arm can lift 0.176Kg (the recommended weight is 0.1Kg), but it is unstable. The motors start to pull on the weighted side but nevertheless still hold.

It may be useful as a helping hand, so long as it's angled in such a way that the weight is near the centre and the object isn't more than double the recommended weight.

Some modifications to make it more stable under load may be required, so that it can perform better under those conditions.

There is an extra disk you can buy which offers activities to do with the arm, but it's more money.

Overall I really enjoyed building this part of the project, and I hope I helped others to have a go at building it too.

Article by Chris "tzj"

Command Line Clinic

By Bobby Redmond (bredman)

Welcome to the Command Line Clinic.

This is the first in a series of articles to help you to start using the command line interface of your Raspberry Pi.

Your computer needs guidance. Without a human to tell it what to do, it will just sit there and sulk. You can click with your mouse, you can prod your finger on a screen, or you can type commands on a keyboard. These are all just ways to tell your computer what to do.

The command line fills new users with dread. It can be very intimidating. Somebody who just clicks icons and buttons with a mouse will expect helpful messages and gentle feedback. A command line spits out gibberish and seldom reports that a command has been successful.

So why do Linux users insist on using the command line? The funny thing is that Linux contains lots of fancy tools that you could use instead of the command line. So why is the command line so popular?

First, the command line gives you the power to do anything you wish. A graphical tool will only support a common task that a lot of people will use. If you need to do something different, you may not be able to find a graphical tool to do the job.

More importantly, you cannot assume that the correct graphical tool will be available when you need it. As an example, have you noticed that the standard Raspberry Pi Debian image has no tool to change the date and time? There are some tools available, but at the moment, these have not been made available for the Raspberry Pi.

Therefore, if you want to set the date and time, you will need to use the command line. This is a good excuse for playing around with commands, because you can't do much damage while messing with the date and time.

So, let's start messing with the command line. You can issue commands just after entering your username and password. Or if you prefer a graphical environment, you can select the desktop main menu item "Accessories" and then "LXTerminal".

Why does Linux use a command line?

Many users meet their first command line when trying out Linux. They wonder why no other operating systems have a command line.

In fact almost every computer operating system has a command line. For example, you can enter commands into Windows by clicking on "Start" then "Run..." and entering the word "cmd". On Mac OSX, press Command+Space and enter the word "terminal".

Your iPhone, your iPad, your Android phone, your modem and your smart TV all have command line interfaces hidden somewhere. For somebody who wants to do something non-standard with their equipment, this command line is the gateway to unlimited access.

Linux is just different because the command line is used more frequently (unlike in Windows) and not hidden completely (like in your iPad).

If you want to see the current date and time, you should enter the command “date”. You should see something like

```
$ date
Fri Jun 1 09:49:25 BST 2012
```

If you wish to change the date/time to June 21 13:45, you could use any of the following commands (whichever is more suitable for you)

```
$ sudo date 06211345
$ sudo date –s “June 21 13:45”
$ sudo date –s “21 June 13:45 2012”
$ sudo date –s “13:45 June 21”
$ sudo date –s “Jun 21 1:45pm”
$ sudo date –s “Next Thursday 13:45”
```

Try experimenting with all the different options which can be used. See what happens when you enter an invalid date such as June 31st.

What does "sudo" mean?

Sudo allows you to change things which will affect the complete machine, not just one user. Sudo is just like “Run as administrator” on a Windows PC. A Linux machine is designed to let the average user work only on his/her own files. To be able to change anything else, the user must add sudo to the beginning of the command. It is a way to ensure that you don’t do something silly unless you really want to.

To see the difference, try entering the date command without sudo at the start. You will get an error “Operation not permitted”.

Some useful commands

Here are some commands for you to experiment with.

`startx`	If you are a bit nervous about the command line, this might be your favourite. It lets you escape to a nice graphical interface.
`free`	Show how much memory is available.
`df /`	Show how much disk space is used.
`hostname -I`	Show your IP address. Try this command with your network cable connected and disconnected to see the difference.
`lsusb`	Show what is plugged into the USB port. Try this command with your mouse connected and disconnected to see the difference.

COMPUTER MUSIC

Interesting facts

The first computer to sing a song was an IBM 7094 in 1961. The song was Daisy Bell (as in 'Daaaisy, Daaaisy, give me your answer, do').

The song was also used in the science fiction film 2001: A Space Odyssey. The intelligent computer Hal 9000 sings the song when he is being deactivated by Dave. The song was the first thing Hal learnt when he was programmed.

If you or a family member own an Apple device with Siri, say to Siri "Sing a song to me" and see what it does.

Back story

It may surprise you to learn that most modern pop music involves the use of one or more computers in its creation. Styles like Hip Hop, Drum & Bass, House, Trance and Dub Step rely heavily on the use of computers. The basic idea is that the computer is programmed to play sound effects in a certain sequence or pattern which becomes a song. The sound effects can be anything from a drum beat to someone singing. So when you press play, the computer itself performs the music as opposed to a band.

The technique is called sequencing. The user will need to have an ear for music though as a bad sequence can sound like a cat's choir. The computer will simply play whatever it's been told to play, nothing more. You could go out and spend a lot of money on professional music equipment to do this, or you could just download free software and do it on your Raspberry Pi for nothing!

What software to use?

You'll encounter a lot of 'my program is better than your program' arguments when you get into making music on computers. My advice is not to limit yourself to any one program, try them all. You may find you are limited to what is available for the Raspberry Pi to start with, but in time this will improve. The program we are going to use in this tutorial is by no means the best program in existance, but it is a good place to start.

The program we are going to use is a tracker program. This is a lightweight program that does what I described above. You load sound effects, called samples, into the program and then you program these sound effects into pages of music, called patterns, which represent the sequence.

To get an idea of the level of music that can be achieved, we're going to get the program running and use it to play back a few songs. Playing back other people's stuff is good fun, I've spent hours doing it before. It will give you a lot of cool ideas and will hopefully spark your imagination.

Introducing Schism Tracker

The program we're going to use is called Schism Tracker.

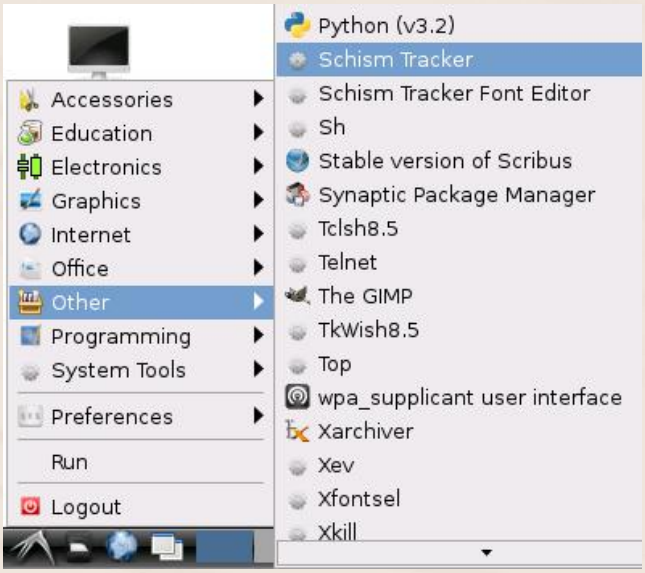
If you are using the recommended Raspbian distribution, there are two ways you can get Schism. The first way is to open a command window and enter the following:

```
$ sudo apt-get install schism
```

The second way is to download Schism from the Pi Store.

Schism Tracker
Schism Tracker is a free reimplementation of Impulse Tracker, a program used to create high quality music without the...
Free!
Open / Download

If you used the apt-get method to install Schism, you will find it in the Application Menu, inside the Other category.



Schism Tracker will start inside a command window. **Note:** avoid maximising the window as this will make Schism Tracker run slowly.

If things are working correctly you should see an image similar to the one below.

Schism Tracker
Using alsa on sdl
Copyright (c) 2003-2012 Storlek & Mrs. Brisby
Based on Impulse Tracker by Jeffrey Lim aka Pulse
Press F1 for copyright and full credits
Continue

Press Enter to continue. Don't be fooled by the old-school look and feel of this program. It is fast, convenient and has survived so long for good reasons.

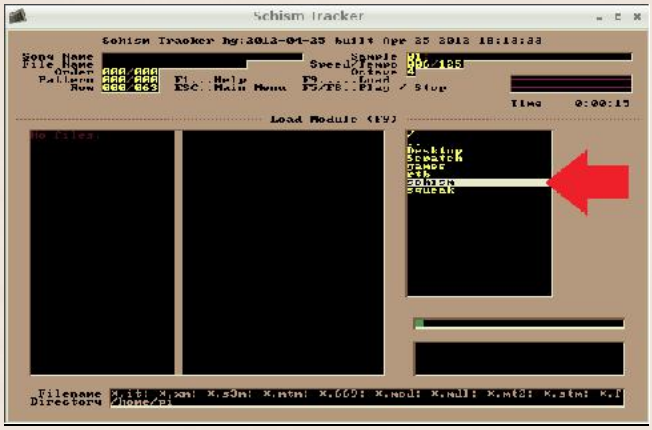
In the world of tracker programs a song file is called a module. To download module files, visit http://modarchive.org where there are hundreds of songs. Use the menu at the top of their site to go to Music... > Charts > Top Favourites - I recommend you listen to the first 20 or so in that list.

[*Ed: To search for specific songs go to Music... > Browse by... > Searching. Some of my favourites are Unreal, Space Debris, Dead Lock, Catch That Goblin, Klisje Paa Klisje, Aces-High2, Kefmania, Funk It Up, Knulla-Kuk and S.C.Out].*

The modarchive.org files come in a compressed zip file, so they need to be extracted before you can play them in Schism.

First create a folder in your Home folder called schism. Download some songs. By default these will appear in your Home folder. Right click each zip file > Extract To... and select the schism folder in your Home folder.

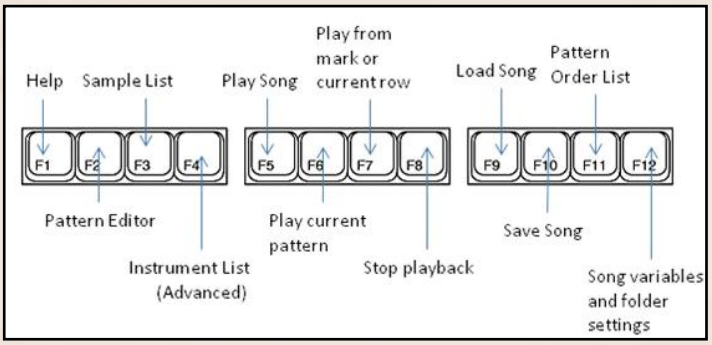
In Schism Tracker, press F9 to load a module. You should see an image like the one shown below.



The right portion of the screen is a folder browser to load module files, you'll need to navigate into the schism folder. You can either use the mouse to double click it or use the cursor keys to move the selection and press Enter.

A list of module files will display on the left portion of the screen. Double click one or use the cursor keys and press Enter to load.

Press F5 to play the song. F8 to stop playback. Press F9 again at any time to load a different module, and F5 again to play it.



This is all we're going to do for this tutorial. In the mean time, just have some fun listening to other people's work.

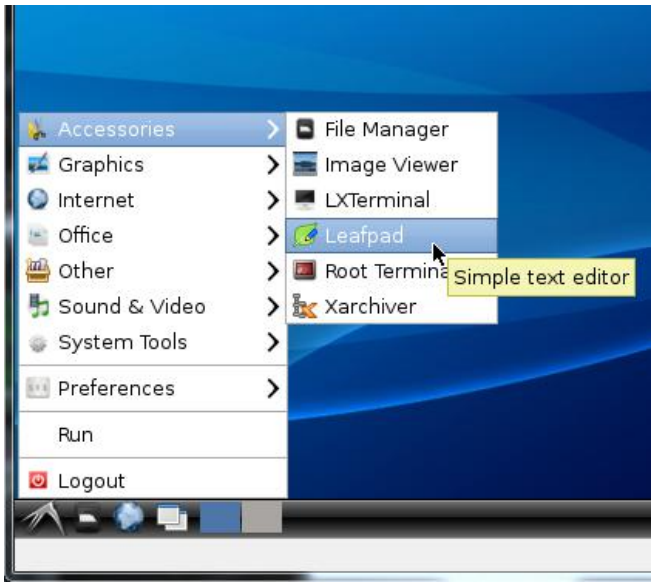
Article by Dave Spice

In issue 1 we showed you how to set up an Oracle VM VirtualBox of the Linux Debian Operating System...

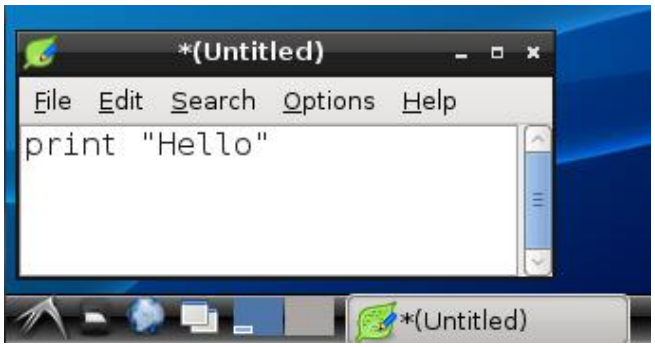
Following on from last time, we thought we would show you how to write and run Python programs using your VirtualBox, and a few simple file and folder operations.

Editing And Running Python Programs

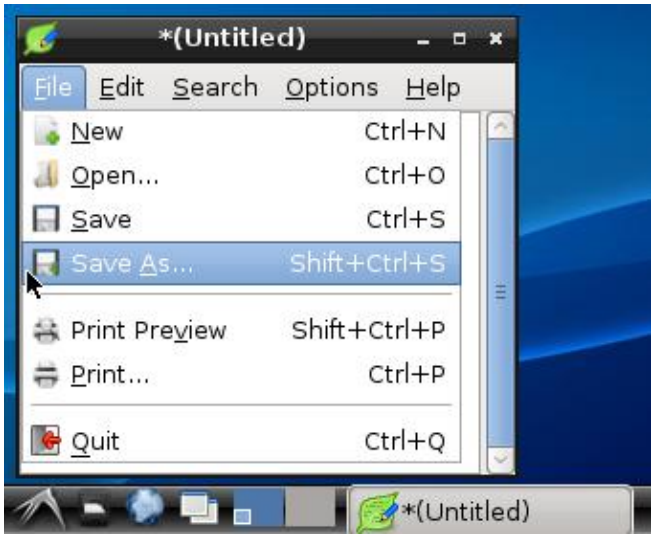
1. Open Leafpad by clicking on the start button and navigate to Accessories > Leafpad.



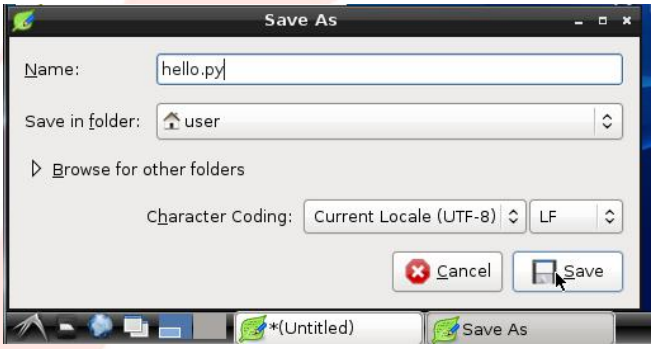
2. Let's type a simple Python program into Leafpad: print "Hello"



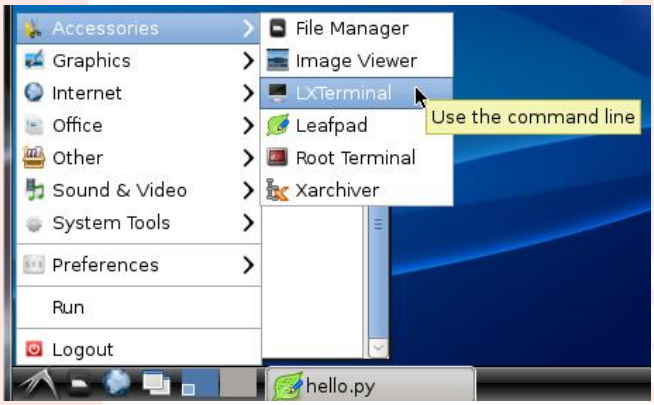
3. Click File>Save As... from the pulldown menu



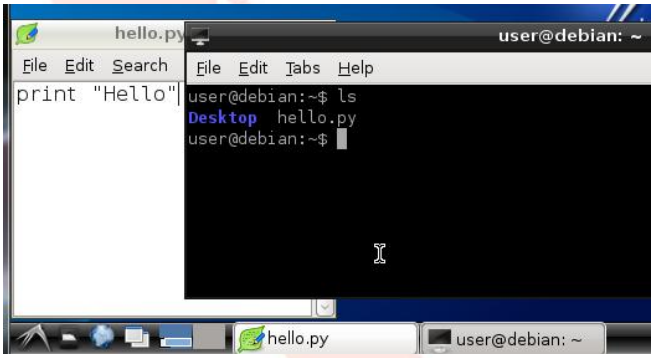
4. Save the program as 'hello.py'.



5. Open LXTerminal by clicking on the start button and navigate to Accessories > LXTerminal.

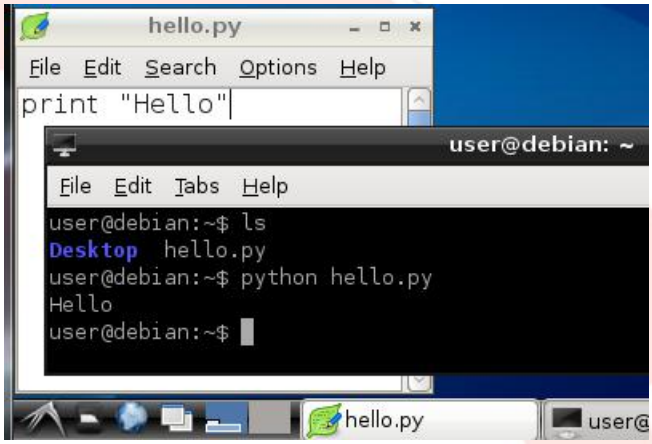


6. In the LXTerminal type 'ls' and press enter to see the list of files in your Home directory. You can think of the Home directory as being the Linux version of 'My Documents'. You should see a directory called 'Desktop' and also your saved hello.py file.

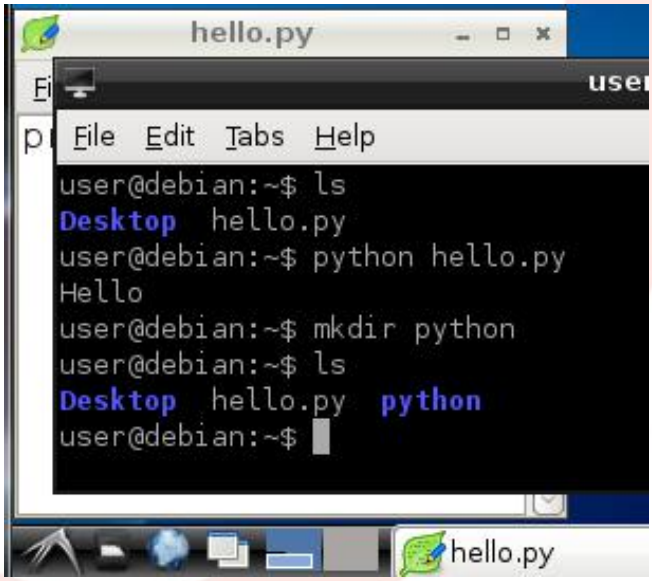


NOTE: The Debian LiveCD comes with Python version 2.6.6. You can confirm this at the LXTerminal prompt by typing 'python -V' (Make sure that is a capital V not lower case v).

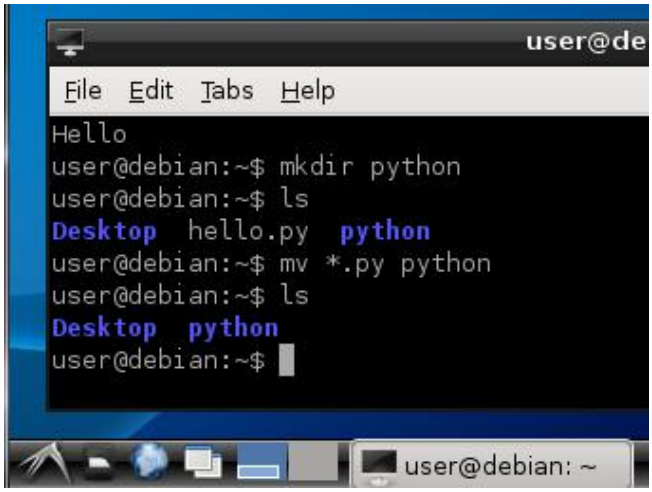
7. Type in 'python hello.py' and press enter to run your Python program. You should see the word 'Hello' printed on the line below.



8. You might like to make a directory / folder to keep your Python programs separate from other files. Type: 'mkdir python'. Then type 'ls' to check that the new directory is there.

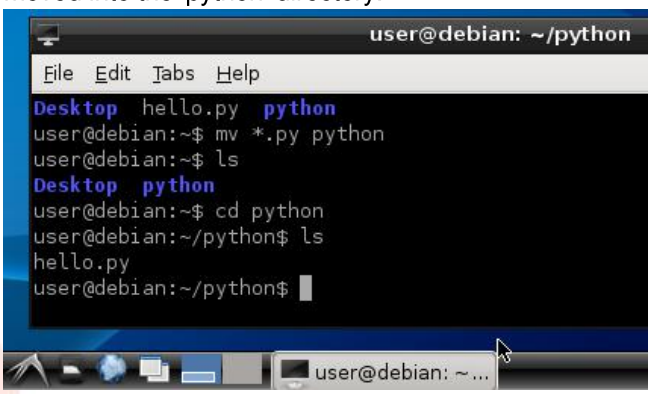


9. Close Leafpad by clicking the X in the top right corner. Then in the LXTerminal type: 'mv *.py python'. Then type 'ls' - you should see that the 'hello.py' file has gone.

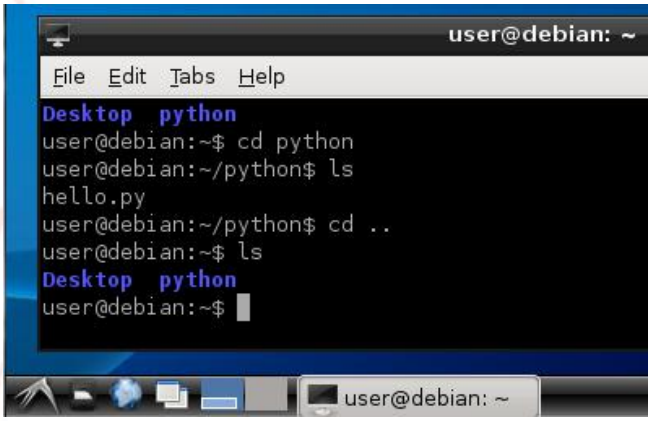


'mv *.py python' - means move all files that end with '.py' into the directory called 'python'.

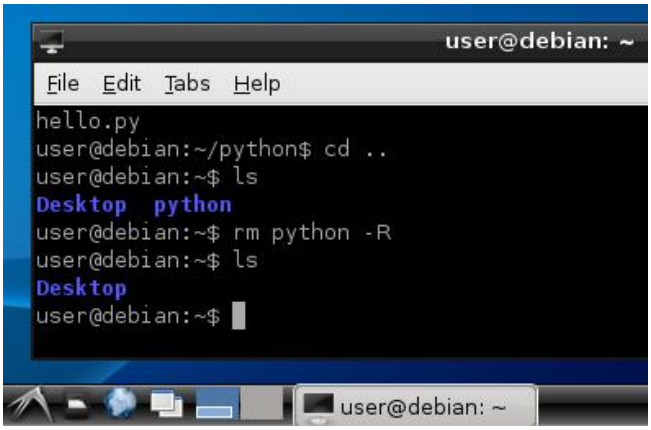
10. Type 'cd python' and then 'ls' to change directory to 'python' and list the files. Make sure that hello.py got moved into the 'python' directory.



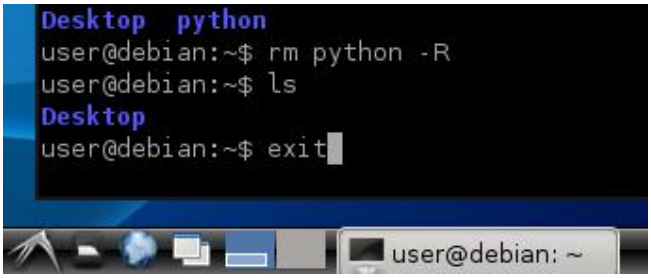
11. Type 'cd ..' to move up one directory level (Back to the Home directory). Type 'ls' again to check where you are. You should see Desktop and python directories.



12. To remove the python directory (delete it) type in 'rm python -R'. Then type 'ls' to see that the python directory is gone. The '-R' means recursive - deletes any files or directories that are inside the python directory as well.



13. To close the LXTerminal type 'exit'.



Well done! Look out for more next month.

Article by Jaseman

the python pit™

Last time we gave you some examples of programs that demonstrate some of the fundamental basics that you will need to know.

This month we are going to start looking at how the Pygame libraries can help to visually improve our programs.

NOTE: For these examples, you will need both Python and Pygame installed on your computer.

We should start by opening a window that is suited to displaying graphical output, rather than just the very simple text output that you get from the terminal or command line:

```
# OPEN A PYGAME GRAPHICS WINDOW
# By Jaseman - 03 May 2012
import os, pygame
from pygame.locals import *
pygame.init()
clock = pygame.time.Clock()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
# This title appears along the top of the graphics window
pygame.display.set_caption("The Title Of My Program")
# Opens a graphics window called 'screen' with width 400 height 200
screen = pygame.display.set_mode([400,200],0,32)
pygame.time.wait(5000) # A 5 second pause before ending the program
```

Try changing the width and height from 400x200 and run the program again to see the results.

PYTHON VERSION: 3.2.2
PYGAME VERSION: 1.9.2a0
O.S.: Win7

TESTED!

When you run this program you should see a black rectangular window appear with a title along the top. All of the pygame graphics will be drawn inside this window. When deciding what size the graphics window should be, consider that some people that use your program might have smaller monitor screens or televisions than you. If you make the window too big, the edges may not be visible on small screens. You also need to think about what your program will be displaying. This will help in deciding the best shape and size for your graphics window.

This time we have added some surface fill commands to paint the entire screen surface in colour. The colour values are defined in amounts of Red, Green and Blue -- we call these RGB colour values.

```
# DISCO SCREEN
# By Jaseman - 10th May 2012
import os,pygame
from pygame.locals import *; pygame.init()
clock = pygame.time.Clock()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
pygame.display.set_caption("Disco Screen")
screen = pygame.display.set_mode([400,200],0,32)
screen.fill((255,0,0));pygame.display.update();pygame.time.wait(2000)
screen.fill((255,255,0));pygame.display.update();pygame.time.wait(2000)
screen.fill((0,255,0));pygame.display.update();pygame.time.wait(2000)
screen.fill((0,0,255));pygame.display.update();pygame.time.wait(2000)
screen.fill((0,255,255));pygame.display.update();pygame.time.wait(2000)
screen.fill((255,0,255));pygame.display.update();pygame.time.wait(2000)
screen.fill((255,255,255));pygame.display.update();pygame.time.wait(2000)
screen.fill((0,0,0));pygame.display.update();pygame.time.wait(2000)
```

The amount of Red, Green or Blue can be any number between 0 and 255 - 0 being very dark and 255 being very bright.

Try experimenting with different amounts of Red, Green and Blue.

You can visit this website which will make it easier to find the right RGB values for the colour you are looking for: **http://www.colorpicker.com**

Move the dot around within the shaded square and drag the arrows to the right up and down. When you find a colour you like, note down the R: G: and B: numbers that are listed to the right hand side. You can ignore the H: S: and B: ones.

PYTHON VERSION: 3.2.2
PYGAME VERSION: 1.9.2a0
O.S.: Win7

TESTED!

Continued over the page...

In this example we draw some basic geometric shapes onto the screen - a circle, a rectangle, a triangle (polygon) and a line.

```
# GEOMETRIC SHAPES
# By Jaseman - 10th May 2012
import os,pygame
from pygame.locals import *; pygame.init()
clock = pygame.time.Clock()
os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
pygame.display.set_caption("Geometric Shapes")
screen = pygame.display.set_mode([400,200],0,32)
# Draw a CIRCLE on screen in (red) at (x,y) coords (60,70) of diameter 40
pygame.draw.circle(screen,(255,0,0),(60,70),40)
pygame.display.update();pygame.time.wait(2000)
# Draw a RECTANGLE on the screen in (yellow) at (x,y,width,height)
pygame.draw.rect(screen,(255,255,0),(70,70,120,60))
pygame.display.update();pygame.time.wait(2000)
# Draw a POLYGON on the screen in (green) at points ((x,y),(x,y),(x,y))
pygame.draw.polygon(screen,(0,255,0),((120,100),(240,40),(220,130)))
pygame.display.update();pygame.time.wait(2000)
# Draw a LINE on the screen in (blue) from (x,y),(x,y),width
pygame.draw.line(screen,(0,0,255),(10,150),(370,30),10)
pygame.display.update();pygame.time.wait(10000)
```

To really get to grips with these new commands, try adding more circles, lines, rectangles and polygons. Change the coordinates and RGB values and see if you can draw a picture.

PYTHON VERSION: 3.2.2
PYGAME VERSION: 1.9.2a0
O.S.: Win7

TESTED!

Notice how the later shapes are drawn on top of the earlier ones. Each shape is drawn in the order that it is listed.

The polygons don't have to be triangles - you can add as many x,y points as you like.

This simple house picture is created using some of the pygame draw commands:

```
# THE HOUSE

# By 0the0judge0 - 29th May 2012

import os, pygame; from pygame.locals import *
pygame.init(); clock = pygame.time.Clock()

os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
d = pygame.draw # this will save us writing 'pygame.draw' many times

pygame.display.set_caption("The House")

# Define some colors
white = (255,255,255); black = (0,0,0)
bg = (186, 213, 48); walls = (157, 109, 9)
door = (151, 36, 9); door_light =(181, 132, 14)

# coordinates of glass panes
windows = [(82, 125), (82, 215), (262, 125), (262, 215)]

#init screen
screen = pygame.display.set_mode([423,347],0,32)

#the walls
screen.fill(bg) #fill the background screen colour
d.rect(screen, black, (60, 102, 305, 225) )
d.rect(screen, walls, (73, 114, 280, 200) )

#the roof
d.polygon(screen,black,((35,112),(121,12),(296,12),(321, 36),(321, 12),
(361, 12),(361, 84),(384, 112)))
d.polygon(screen,walls,((62,101),(128,23),(289,23),(334, 69),(334, 25),
(348, 25),(348, 88),(361, 101)))

#the door
d.rect(screen, black, (167, 198, 84, 125 ))
d.rect(screen, door, (179, 210, 60, 101 ))
d.rect(screen, black, (185, 216, 50, 54 ))
d.rect(screen, door_light, (191, 222, 38, 41 ))
d.circle(screen,black,(209,277),5)

#the windows
for window in windows:
      d.rect(screen, black, (window[0], window[1], 76, 76) )
      d.rect(screen, white, (window[0] + 12, window[1] + 12, 22, 22) ) #tl
      d.rect(screen, white, (window[0] + 42, window[1] + 12, 22, 22) ) #tr
      d.rect(screen, white, (window[0] + 12, window[1] + 42, 22, 22) ) #bl
      d.rect(screen, white, (window[0] + 42, window[1] + 42, 22, 22) ) #br

#let's see the end result
pygame.display.update()
pygame.time.wait(10000)
pygame.quit())
```

PYTHON VERSION: 3.2.2
PYGAME VERSION: 1.9.2a0
O.S.: Win7

TESTED!

Feedback

Nice concept, nice layout, nice graphics, nice mag overall.

Dave_G_2

Looks great, congrats. I can't wait for the next one.

kirbyman62

Hey, I love the design of the mag – very pro.

pifan

Just had a scan through the MagPi. It looks really good; seems like a great deal of thought and effort has gone into it.

budgieboy

Really like what you did for a first issue!

Rek

Excellent! I enjoyed reading it. Now just waiting to hear from RS in South Africa when I can move from the waiting list to "ordered".

Kaspaas

Very attractive. I've just skimmed it so far, but my overall impression is that it is very professionally done. kudos.

boley

Just read the mag and its fascinating. VERY well done.

secretreeve

Absolutely excellent – much better than I expected. In particular the Python Pit is just what beginners need.

bredman

Hi there, congrats with your first issue – well done!

teh_orph

What a terrible site. Sorry. When I finally manage to get the PDF and read it in xpdf, it's OK, but rather primitive.

GordonH

I quickly read thought it and was pleasantly surprised. Its well laid out, hits the target markets (nice child friendly areas for coding) and has some nice content.

JoeDaStudd

Thanks very much for providing this mag. I'm very much looking forward to further issues and to see how it develops.

Forrrge

Well done. You've done a good job.

trikidiki

It's a great start; one hell of a lot better than I could have done.

rurwin

Thanks for a great first issue. Personally I liked the PDF version and I printed out and stapled together a few copies and left them around the staffroom of the school I work at. They got a lot of interest. Keep up the great work!

rubikwizard

You have done something amazing here. I'm 15 and I must say I buy magazines for a fiver with less engrossing content than this. I found every article interesting, and after showing my friends they too had a great read.

FullXion

Kudos on the first issue. Very well put together for a first issue, and i'm sure if the momentum keeps going there will be some interesting stuff in there as you cover peoples projects and suchlike.

ArborealSeer

While I didn't learn anything groundbreaking (I'm a software developer) I did learn a few new things about Linux and think the coding tutorials are pitched at the perfect level for a beginner! Well done guys!

Tavalin

Image from photofunia.com

The MagPi™

Raspberry Pi is a trademark of the Raspberry Pi Foundation. The MagPi magazine is collaboratively produced by an independent group of Raspberry Pi owners, and is not affiliated in any way with the Raspberry Pi Foundation. The Magpi does not accept ownership or responsibility for the content or opinions expressed in any of the articles included in this issue. All articles are checked and tested before the release deadline is met but some faults may remain. The reader is responsible for all consequences, both to software and hardware, following the implementation of any of the advice or code printed. The MagPi does not claim to own any copyright licenses and all content of the articles are submitted with the responsibility lying with that of the article writer.

Tandy have generously provided The MagPi with a Domain name.

A great big 'THANK YOU' from The MagPi Team.

editor@themagpi.com

Other Resources and Weblinks

http://www.themagpi.com
Official website of The MagPi magazine.

http://www.raspberrypi.org
Official home of the Raspberry Pi Foundation.

http://www.raspberrypi-spy.co.uk
Raspberry Pi resources, tutorials, help and downloads.

http://www.raspberrypiusers.com
A Raspberry Pi website/blog.

http://scratch.mit.edu
Official Scratch website.

http://www.linuxuser.co.uk/tutorials/how-to-set-up-raspberry-pi
Liam Fraser's article on setting up your Raspberry Pi (SD cards).

http://www.designspark.com/theme/raspberrypi
RS Components online resources and design support for Engineers.

The MagPi Issue 02 JUNE 2012

Team:

Ash Stone
Chief Editor / Administrator / Header

Jason 'Jaseman' Davies
Editor / Page Designs / Writer / Website

Tim 'Meltwater' Cox
Writer / Editor / Photographer / Page Designs

Chris 'tzj' Stagg
Writer / Editor / Photographer / Page Designs

Bobby 'bredman' Redmond
Writer / Page Designs

Darren Grant
Writer / Page Designs

0The0Judge0
Python Tutor / Administrator

Dave Spice
Writer

Print Edition Preparation:

W.H.Bell, Ian McAlpine, Colin Deady, Aaron Shaw, Isa McKenty, Simon Johnson, Alan 'paisleyboy' Holroyd, Sam Marshall, Joshua Marinacci, Colin Norris, Mark Robson, Steve Drew, Alex Baker

Raspberry Pi is a trademark of the Raspberry Pi Foundation.